Set-theoretic graph rewriting

Jean-Claude Raoult, Frédéric Voisin

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

1992 25ème anniversaire

N° 1665

Programme 2
Calcul Symbolique, Programmation
et Génie logiciel

SET - THEORETIC GRAPH REWRITING

Jean-Claude RAOULT
Frédéric VOISIN

Avril 1992

RR_1665

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTEMES ALEATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CEDEX FRANCE
Tél. : 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

SET - THEORETIC

GRAPH REWRITING

RECRITURES DE GRAPHES

ENSEMBLISTES

Publication Interne n°648 - Mars 1992 - 18 pages

Programme 2

Jean-Claude Raoult
IRISA, Campus de Beaulieu
F-35042 RENNES CEDEX

Frédéric Voisin
LRI, Bâtiment 490
F-91405 ORSAY CEDEX

Abstract: Considering graphs as sets of vertices and arcs, we define their rewritings as simple set-theoretic rewritings: applying a rule consists of removing the left-hand side and adding the right-hand side. This point of view is explored through some instances. We show that these rewritings can simulate conventional graph rewritings, at least those which do not involve a quotient, and term rewritings, at least those in which no left nor right-hand side is reduced to a variable. The simplicity of the approach is illustrated by a criterion of local confluence analogous to the well-known Knuth & Bendix criterion.

Résumé : *De considérer un graphe comme un ensemble de sommets et d'arcs permet de définir leurs récritures comme de simples récritures d'ensembles : on ôte le membre gauche et on ajoute le membre droit. On étudie ce point de vue en donnant quelques exemples et en simulant les récritures de graphes classiques, du moins celles qui ne font pas intervenir de quotient, et les récritures de termes, du moins celles où aucun membre gauche ni droit n'est réduit à une variable. On illustre la simplicité de cette approche en établissant un critère de confluence locale analogue au critère de Knuth & Bendix.*

---

26 March 1992

SET-THEORETIC

GRAPH REWRITING

Jean-Claude Raoult
IRISA, Campus de Beaulieu
F-35042 RENNES CEDEX

Frédéric Voisin
LRI, Bâtiment 490
F-91405 ORSAY CEDEX

I. INTRODUCTION

GRAPH rewriting is by now an old story: Schneider's first definitions, for instance, date back to 1970. See also the survey by Nagl [1979]. The theory was at first rather involved, and ever since people have tried to simplify the method. Compare for instance the definition cited above with a later one given by the same author (cf. Ehrig, Pfender & Schneider [1973]) under the name of "algebraic graph grammars". This last method is widely known, thanks to the numerous applications published by the Berlin school, even if it is really applicable only in a restricted case, that of "fast productions". This remark has been made by Raoult [1984] who generalized the construction, and simplified its use in the proofs. This last version has been simplified yet and bettered by Kennaway [1987] and Löwe & Ehrig [1990]. During the same time, the theory of term rewriting has been developing rapidly. Rewritings in free algebras have been known fairly early and criteria for confluence and termination have got more and more precise (cf. for instance the survey by Dershowitz [1985] or Rusinowitch [1989]). In quotient algebras, the same problems have been tackled with some success (cf. Kirchner [1985]). By now, terms can be rewritten which contain associative operators, or associative and commutative, or associative, commutative and idempotent; sets of terms, for instance, can be rewritten.

The goal of the present work is to connect graph rewritings to simple set rewritings, following the intuitive notion of a graph being a set of edges, or rather hyper-edges. This idea has already been exploited by Bauderon et Courcelle [1987]. They describe (hyper) graphs with natural numbers for vertices, and constructors on sequences of numbers (i.e. hyper-edges): mainly the disjoint union and the quotient, with explicit renumberings of vertices. A given hyper-graph is represented by a term on these operators, or rather several equivalent such terms. The equivalence is described by a recursive set of axiom schemata,

which turns the set of hyper-graphs into an effectively presented algebra. We develop here a slightly different point of view, following Raoult [1985]: vertices will be variables, of which we have a countable reserve — as in the case of first order terms. A hyper-edge is now the theoretical analogue of a statement in a three adress code, in which the function $f(x, y)$ is associated with a variable z containing the result of the computation of $f : z = f(x, y)$. This will be denoted by the hyper-edge $fzxy$. In this setting, a hyper-graph is simply a set of hyper-edges, while a program written in this code is a sequence of hyper-edges. If each variable is assigned only once, both concepts coincide.

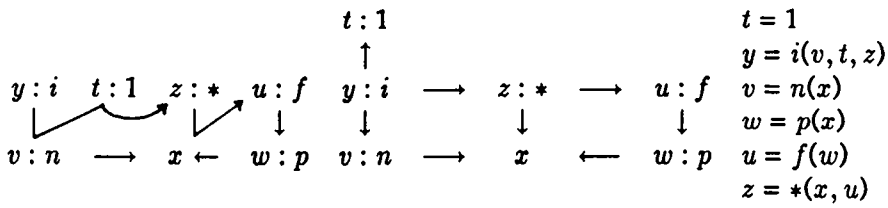


Figure 1. Three representations of the same hyper-graph. On the left is a hyper-graph, in the middle a graph, on the right a sequence of three adresss statements. What do they describe?

Further, we shall avoid identifications: our thesis is that a rewriting is a "free" operation, so to speak, and identifying two vertices is not a rewriting but a quotient. This is closer to term rewritings and also to code optimisations, where identifications are seldom performed, for reasons of undecidability. An important consequence is a property of rewritings which we feel characteristic: they are invertible, like term-rewritings: just swap both sides of each rule.

In section II, we describe free rewritings of sets of hyper-edges. Actually the same method allows the rewriting of multiple hyper-edges, with no more fuss, possibly less. In section III, we compare these rewritings with the more or less standard approaches for graph rewriting; and in section IV we discuss the simulation of term rewritings by hypergraph rewritings. Section V is a conclusion.

II. FREE REWRITINGS

Let X be a countable set of variables x, y, z, etc, and let F be a finite set of "function symbols", which will label the hyper-edges.

Definition 1. *A hyper-edge is an element of the set* FX*, *and a hyper-graph is a set of hyper-edges included in* FX*. *The set of variables occurring in* H *is denoted by* X(H).

Function symbols are usually considered together with an integer arity $\alpha : F \to \mathrm{N}$ and the hyper-graphs satisfy the condition that all words $fx_1 \ldots x_n$ have length $\alpha(f)+1$. It is no more and no less difficult to put syntactic type conditions, with a signature. We shall do without it.

The union H ∪ H′, intersection H ∩ H′ and difference H − H′ of two hyper-graphs H and H′ are defined as for any subsets,or multisets, of FX* , but the union will often be denoted by H + H′. If we write singletons without curly brackets, hyper-graphs can be written like series with boolean coefficients:

$$\mathrm{H} = 1t + iyvtz + nvx + pwx + fuw + *zxu$$

(cf.figure 1). If the coefficients are natural numbers, the hyper-graphs may have multiple hyper-edges (cf. example 4 below): they become multisets of hyperarcs.

A hyper-graph morphism is a mapping σ:FX* → FX* extending a mapping σ:X→X. It is a particular case of a substitution where the image of a variable is a variable: $\sigma(x) \in X$. As for substitutions, the domain of σ, dom(σ), is the set of variables that differ from their images. If the mapping is one-to-one (then there exists a left inverse σ^{-1}), it is called a vertex renaming. The corresponding classes of isomorphism correspond to Bauderon & Courcelle's abstract graphs as opposed to concrete graphs with given names for vertices. The only substitutions considered in this setting will be from variables to variables.

Beware that graph isomorphism is not preserved by adding an arbitrary context, as in the following simple example pointed to us by Detlef Plump.

Example 1: the graphs ax and ay are clearly isomorphic. Adding the context bxy yields $bxy + ax$ and $bxy + ay$ which are not isomorphic! Only graph equality is preserved.

In the following definitions, the symbol + is used in order to handle both set and multiset union. Rewriting systems are simply sets of couples of hypergraphs: relations over $\mathcal{P}$(FX*), the set of hypergraphs.

Definition 2. *One step of rewriting generated by a relation* R *over hypergraphs is the relation* H → H′ *which is true if*

(*i*) H = C + G *is a partition of H, and* H′ = C + D; *if*

(*ii*) X(G) ∩ X(C) ⊇ X(D) ∩ X(C) *and if*

(*iii*) (G, D) ∈ R *up to vertex renaming.*

Since the variables occurring in the rule are renamed before the rule is embedded into a context, two rules differing only by the names of the variables generate exactly the same relation.

Remark 1: If A → B generates G → H then Aα → Bα generates G → H for all variable renaming α.

The relation (*ii*) may also be written as [X(D) − X(G)] ∩ X(C) = ∅ stressing the fact that the "new" variables appearing in D must really be new, even in H′. The result of the generated relation is in fact defined up to a renaming of these variables, as expressed by the following remark.

Remark 2: Let A → B be a rule generating G → H and α be a variable renaming. Then A → B generates Gα → K and H is isomorphic to K for a renaming of variables outside X(Hα))

Note that we need not bother about "dangling arcs". If a vertex occurs in a left-hand member, and not in the right-hand member, then either it occurs also in the context graph, in which case it remains in this same context graph after rewriting, or it does not occur elsewhere, and disappears in the result.

Example 2: The rule $bxy + byz \rightarrow bxz + byz$ applied to $bxy + byz + bxu + buy + buz$ yields $bxz + byz + bxu + buy + buz$. This is a case where the variable y does not disappear. The rule describes multiple jumps elimination in optimizing compilers, where b stands for "branch" and x and y are the labels of the source and target instructions.

Nevertheless, variables in G − D occur less frequently in the result. This poses problems in the case where hypergraphs represent terms as in the following example, in which variable y disappears in the right-hand side of the rule:

$pxy + syz \rightarrow Ixz$, applied to $pxy + syz + auxy + 0z$ yields $Ixz + auxy + 0z$. Now if pxy, syz and Ixz mean respectively that x is the predecessor of y, that y is the predecessor of z, and that x is equal to z, the first member means that $u = 0 + 1$ while the result means $u = 0 + y$ (?).

This would not happen, should the rule satisfy the more symmetric condition X(G)=X(D) (Actually this condition could be required only on the occurrence: $X(G) \cap X(C) = X(D) \cap X(C)$). A rule satisfying this condition will be called *reversible*. The following result is obvious.

Fact: if $H \rightarrow K$ by a reversible rule $G \rightarrow D$, then $K \rightarrow H$ by the reversible rule $D \rightarrow G$.

There is an easy extension of the definition allowing a restricted form of quotient.

Definition 3. *One step of rewriting generated by a relation R over hypergraphs is the relation* $H \rightarrow H'$ *which is true if*

(i) $H = C + G\sigma$ *is a partition, and* $H' = C + D\sigma$, *if*

(ii) $X(G\sigma) \cap X(C) \supseteq X(D\sigma) \cap X(C)$ *and if*

(iii) $(G\sigma, D\sigma)$ *is deduced from* $(G,D) \in R$ *by changing the name of the variables in* G, *possibly identifying some of them:* $dom(\sigma) \subseteq X(G)$.

In fact, this definition is not broader than the previous one: it is always possible to close any given relation R over hypergraphs so that the closure contains all pairs deduced from a given pair (G,D) by all possible quotients $(G\sigma, D\sigma)$ for $dom(\sigma) \subseteq X(G)$, and up to variable renaming. If R is finite, the closure is also finite, and applying Definition 2 to this closure of R is equivalent to applying Definition 3 to R itself. In this case, R appears as a rule schema, and $(G\sigma, D\sigma)$ is an instance of a rule. Henceforth, we shall suppose the first definition, which allows finer rewritings. If needed, we shall assume that we have in our system all instances of the rule schemata.

Note that we have just defined a rewriting on terms over $F \cup \{+\}$ where + is associative, commutative and idempotent (only associative and commutative if we allow integer coefficients: multisets), but restricted to a subset of simple terms: terms of depth one.

Example 3: The following system in which $ixuyz$ represent the conditional (if u then x equals y, else x equals z)

$$ixuyz + \wedge uab \rightarrow ixavz + ivbyz$$
$$ixuyz + \vee uab \rightarrow ixayv + ivbyz$$
$$ixuyz + \neg ua \rightarrow ixazy$$

describes the replacement, common in compilation, of a boolean expression by a sequence of elementary tests.

Example 4: The following system with integer coefficients:

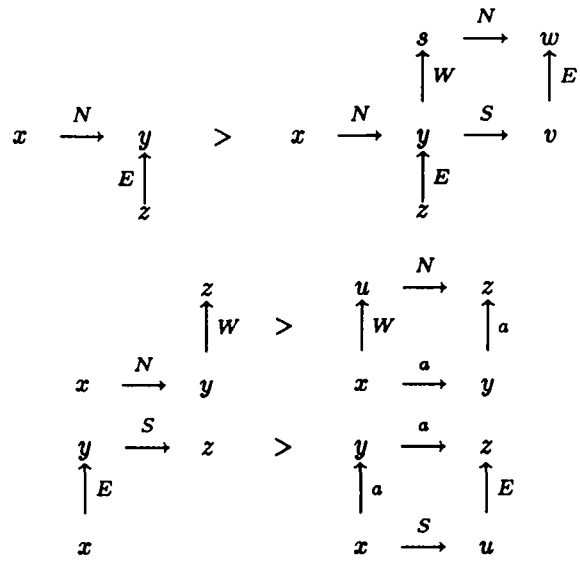
$$axy \rightarrow axz + azy$$
$$axy \rightarrow axy + axy = 2axy$$

generates all series-parallel graphs when started with a graph with a unique arc. This system is context-free. For hyper-edge rewritings, see Habel & Kreowski [1987].

Example 5: The following last example generates $\mathbb{N} \times \mathbb{N}$ starting from a square $axy + axz + Eyu + Nzu$. Arcs labelled N, S, E, W are the non-terminals on the northern, southern, eastern and western sides of each square:

$$Nxy + Ezy \rightarrow Nxy + Ezy + Wys + Syv + Evw + Nsw$$
$$Nxy + Wyz \rightarrow axy + ayz + Wxu + Nuz$$
$$Exy + Syz \rightarrow axy + ayz + Sxu + Euz$$

These rules are visualized below.



Single hyper-edge replacement systems generate graphs having a bounded degree of connectivity and therefore cannot generate this grid. (cf. for instance Habel & Kreowski [1987] or Sopena [1987]). This system is therefore strictly stronger.

III. RELATION WITH THE "ALGEBRAIC GRAPH REWRITINGS"

There are several ways in which graphs may be translated into hypergraphs. One of them has been used informally in example 2, and is suitable for graphs with labelled arcs and vertices: an arc $x \to y$ having label f is easily translated into the word fxy, and the fact that vertex x is labelled with a is rendered by ax. This view stresses the fact that graphs are just particular cases of hypergraphs.

Since our formalism does not allow identification of distinct vertices, it is not possible to simulate the general form of graph rewriting described by the Berlin school, using two push-outs (cf. Ehrig & al. [1974]) — nor *a fortiori* the general form using a single push-out technique described by Raoult [1985] and improved by Kennaway [1987], and Löwe & Ehrig [1990].

Definition 4. *Let* G *be a graph with set* V *of vertices labelled in* S *and set* A *of arcs labelled in* F. *The canonical hypergraph* h(G) *associated with* G *is the set*

$$h(G) = \{ax; x \in V \wedge x \text{ has label } a \in S\} \cup \{fxy; \exists (x \to y) \in A \text{ labelled by } f \in F\}$$

This corresponds to associating variables with vertices. Usually, the vertices are identified with addresses in memory and the correspondence between addresses and variables is standard.

Proposition 5. *Suppose that in the fast production* G ← K → D *both morphisms are one-to-one and* K *consists only of vertices. Then a graph* L *rewrites to* R *according to the fast production only if* h(L) → h(R) *for the rule* h(G) → h(D).

Proof: Call i and j the injective morphisms of K into G and D. Since i and j are one-to-one, we shall identify K with a subset of vertices of G and of D. Thus in the correspondence h, K is identified with $X(h(G)) \cap X(h(D))$. Let $g : G \to L$ be an occurrence of the left-hand side of the rule, and C be the push-out complement:

$$C = L - g(G) + g(K)$$

Then the push-out R is defined by $R = (C + D)/K$ where the quotient indicates that we must identify each vertex $x \in D$ with the vertex $g(x)$ of C.

$$\begin{array}{ccccc} G & \xleftarrow{i} & K & \xrightarrow{j} & D \\ {\scriptstyle g}\downarrow & & \downarrow & & \downarrow \\ L & \longleftarrow & C & \longrightarrow & R \end{array}$$

Define a substitution σ by

$$\begin{cases} \sigma(x) = g(x) & \text{if } x \in h(G) \\ \sigma(x) = x & \text{otherwise} \end{cases}$$

Then $h(R) = h(L) - (G) + h(D)$. Instead of identifying the vertices of K in R with those of $g(K)$ we take off those of $g(K)$ and put them back with D. The equality means that the result of the graph rewriting step corresponds to the result of the rewriting step of the associated hypergraphs, QED.

Note however that in the case of graphs, a condition for applying the graph rewriting is (cf. Ehrig [1987]):

$$\text{DANGLING} \cup \text{IDENTIFICATION} \subseteq K$$

where

$$\begin{aligned} \text{DANGLING} &= \{x \in G; (\exists a \in G - K)\ \text{source}(a) = x \ \vee\ \text{target}(a) = x\} \\ \text{IDENTIFICATION} &= \{x \in G;\ (\exists y \in G)\ x \neq y \ \wedge\ g(x) = g(y)\} \end{aligned}$$

In our case these conditions are by no means necessary, as shown by the following example.

Example 6: Let R be the unique rule $axy + bxz \to cxz$. Apply it to the (hyper)graph $G = auy + axy + bxz$. We get $H = auy + cxz$. Here y is dangling. Apply it to the (hyper)graph $G = auy + axy + bxy$. We get $H = auy + cxy$. Here y is at the same time dangling and identified.

IV. SIMULATION OF TERM REWRITINGS

In this section graphs are considered to be shared representations of terms, and graph rewriting an optimization of term rewriting, as for example in Raoult [1985], Barendregt & *al.* [1987] or Habel & *al.* [1987]. The following is therefore relevant.

Definition 6. *We shall consider only acyclic graphs with the additional requirement that each vertex must have "single assignement": for each vertex v there is at most one hyperarc fvw.*

Such an arc will be considered as directed, with v being the source. We shall also distinguish a vertex, the root of the graph, and only those vertices that are reachable from the source of the graph really matter. Other vertices are subject to garbage collection.

Let H be an acyclic hypergraph with single assignment and x be a variable in X(H). Define a function exp : $(H, x) \mapsto \exp(H, x) \in T(F, X)$, the term algebra over (F,X) by:

$\exp(H, x) = f(\exp(H, x_1) \ldots \exp(H, x_n))$ if there exists $f x x_1 \ldots x_n$ in H, and
$\exp(H, x) = x$ otherwise.

Hence, exp unfolds a graph into a term, starting from a given root vertex. Notice that the arity of a function symbol f is one less than the arity of this same symbol considered as a label for a hyperarc. We will suppose that such a root vertex is fixed and we shall usually omit to mention it explicitly. We shall find two right inverses for exp as in Raoult [1984], according to whether we want a minimal or a maximal sharing of identical subterms.

Notations: In the rest of this paper, following the Handbook[1990] we consider a term t as a function the domain dom(t) of which is the subset of all its occurrences (positions): for each $u \in \mathrm{dom}(t)$, we denote by t/u the subterm of t at the occurrence u and by $c(u) \in F \cup X$ the root of this subterm. Moreover, $c[t]$ denotes the term c with t grafted at one of its occurrence: $(\exists u \in \mathrm{dom}(c))\ c[t]/u = t$. Finally X($t$) denotes the set of the variables of t.

Least contracted representation: Let t be a term in T(F,X). With each occurrence $u \in \mathrm{dom}(t)$ we associate a variable:

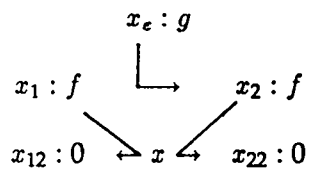
$$\begin{cases} u \mapsto t(u) & \text{if } t(u) \in X, \\ u \mapsto x_u & \text{otherwise.} \end{cases}$$

If t is reduced to a variable, no graph is associated with t. Else define the following hypergraph:

$$G(t) = \{ f x_u x_{u1} \ldots x_{un};\ t/u = f t_1 \cdots t_n \}$$

The root of the graph is x_ϵ, the (new) variable associated with the empty occurrence. In this construction, the sharing involves variables and variables only, as shown by the following example:

Example 7: The least contracted graph corresponding to the term $t = g(f(x,0), f(x,0))$ is



Or, linearly and after a suitable renaming: $G(t) = guvw + fvxy + fwxz + 0y + 0z$

Fact: For all term t we have: $\exp(G(t), x_\epsilon) = t$.

Proof: By induction on the structure of t, QED.

Note that we cannot represent any term by a hypergraph, that is a set of hyperarcs: since vertices are not dealt with independently of arcs, we have no means to represent terms having no arc: variables. Therefore, we exclude terms reduced to a variable!

Proposition 7. *Let $t = c[t']$ be a term with a sub-term t', then* $G(t) = H + H'$ *where + denotes the set union and*

(i) $H = G(c[x_u])$ *where* $t/u = t'$, *and*

(ii) H' *is deduced from* $G(t')$ *by the renaming* $x_v \mapsto x_{uv}$; *hence* $X(H) \cap X(H') = \{x_u\}$.

Proof: Let u be the occurrence in dom(t) where t' is grafted. Replacing t' by x_u in t we get a term $c[x_u]$ having the associated graph H. We have dom(t) = dom($c[x_u]$) + $u$dom(t'). Then for any $v \in \text{dom}(t') \Leftrightarrow uv \in \text{dom}(t)$, and $t'(v) = t(uv)$. This implies that any hyperarc in G(t') appears in G(t), under the vertex renaming $\sigma(v) = uv$. Moreover t and c coincide on dom(c) hence the equality, QED.

There is a similar decomposition for substituted linear terms:

Proposition 8. *Let $s[x_1, \ldots, x_n]$ a term linear in $\{x_1, \ldots, x_n\}$ and σ a substitution such that $x_i\sigma \notin X$ for $i = 1, \ldots, n$. Then* $G(s\sigma) = H_0 + H_1 + \ldots + H_n$ *where*

(i) $H_0 = G(s[x_{u_1}, \ldots, x_{u_n}]$ *where* $s/u_i = x_i$ *for* $i = 1, \ldots, n$.

(ii) H_i *is deduced from* $G(x_i\sigma)$ *by the variable renaming* $x_v \mapsto x_{uv}$ *for all occurrence v of $x_i\sigma$ which is not a variable.*

Proof: Since s is linear, for each x_i, $1 \leq i \leq n$, there exists a unique occurrence u_i in dom(s) such that $s/u_i = x_i$ and we have a partition of dom($s\sigma$):

$$\text{dom}(s\sigma) = \text{dom}(s) + \sum_i u_i \cdot \text{dom}(x_i\sigma) = \text{dom}(s) + \sum_{1 \leq i \leq n} D_i$$

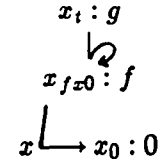
where $D_i = \text{dom}(s\sigma) \cap u_i\omega^*$. Define a mapping α_i from dom($x_i\sigma$) to D_i by $\alpha_i(v) = u_iv$. Then $(s\sigma)/\alpha_i(u) = x_i\sigma(u)$ for all $u \in \omega^*$. Therefore each arc in G($x_i\sigma$) appears in G($s\sigma$) under the vertex renaming $x_u \mapsto x_{\alpha_i(u)}$. Since $s\sigma$ and s coincide on dom(s) $- \{x_u\}$, we have the proposition, QED.

Note that the condition on σ is used here: take $s = f(x, y)$ for which $H_0 = fx_exy$ and $\sigma = [y/x]$. Then $s\sigma = f(x, x)$ with corresponding graph fx_exx which is deduced from H_0 by a mapping of variable (not a renaming). It is not decomposed into a part corresponding to s and another part corresponding to σ.

The most contracted representation: We shall now define a representation sharing as many subterms as possible. With a term t we associate a hypergraph C(t) defined as before except that we now associate a variable with each subterm t' of t: if t' is a variable then use t' else use a new variable, say $x_{t'}$. The root of the graph is x_t, the variable associated with t itself. Then set:

$$C(t) = \{fx_{ft_1 \cdots t_n} x_{t_1} \cdots x_{t_n};\ (\exists s)t = s[ft_1 \cdots t_n]\}$$

Example 8: For the term $t = g(f(x, 0), f(x, 0))$ of the previous example, C(t) is the following hyper-graph:



Or, after renaming: $C(t) = guvv + fvxw + 0w$.

Fact: For all terms t, we have: $\exp(C(t), x_t) = t$.

Proof: by induction on the size of t, QED.

Representation of term rewriting rules: We already mentioned that we cannot represent terms collapsed to variables. Therefore there will be no straightforward translation of "collapsing" rules like $spx \rightarrow x$ (we shall return to this problem at the end of this section). However, where the left-hand side of a rule is not a variable, such a translation of terms rewriting rules by graphs rewriting rules is possible and, in general, there can be several such rules which differ by the graphs used in both sides of the rule, and by the way vertices are named. A standard representation of rules must be adopted, for graphs rewriting to implement terms rewriting faithfully: the least contracted representation is a more natural choice for this purpose. We now define the hypergraph rewriting rule to be associated with a given term rewriting rule $s \rightarrow t$.

Definition 9. *Let $g \rightarrow d$ be a term rewriting rule in which neither g nor d are variables and s is linear. Then $G(g) \rightarrow G(d)$ is the associated hypergraph rewriting rule, where $G(g)$ and $G(d)$ are as defined above, except that the only variables in $X(G(g)) \cap X(G(d))$ are the variables of $X(g) \cap X(d)$ and the variable associated with their roots:*

$$X(G(g)) \cap X(G(d)) = X(g) \cap X(d)) \cup \{x_\epsilon\}.$$

The interface between $G(g)$ and $G(d)$ is restricted to contain only their common root, named x_ϵ, and the term variables: $X(g) \cap X(d)$. All other vertices of $G(d)$ must be new, to prevent the target graph from being rewritten outside of the instance of the left-hand side. Note also that term variables are preserved in our graph representation: a variable in $X(g) \cap X(d)$ is not the source of any arc in $G(g)$ or $G(d)$. Therefore rewriting of an (acyclic) graph with the single assignment property by a rule $G(g) \rightarrow G(d)$ always yields an (acyclic) graph with the single assigment property. Finally note that if g is linear, there is no sharing in $G(g)$ even at the variables: the graph $G(g)$ is a tree.

Theorem 10. *Let $g \rightarrow d$ a term rewriting rule in which g and d are linear, and neither is a variable. If $r \rightarrow r'$ for this rule, there exists a hypergraph H such that $G(r) \rightarrow H$ under the deduced hypergraph rewriting rule and $G(r')$ is the sub-hypergraph of H reachable from the root of H, up to variable renaming.*

Proof: If $r \rightarrow r'$, there exists a context c, possibly empty, and a substitution σ such that $r = c[g\sigma]$ and $r' = c[d\sigma]$. The propositions proved above show the following decomposition of $G(r)$ with the suitable proviso concerning variable renaming:

$$G(r) = G(c) + G(g\sigma) = G(c) + G(g) + \sum_{x \in dom(\sigma)} G(x\sigma)$$

This graph rewrites into:

$$G(c) + G(d) + \sum_{x \in dom(\sigma)} G(x\sigma) = H$$

Now either c is not empty in which case both $G(r)$ and H are rooted by the root of $G(c)$, or c is empty and both graphs are rooted by their common renaming of vertex x_ϵ of $G(g)$ and $G(d)$. In either case the vertices of H which are reachable from its root are the vertices of $G(c)$, $G(d)$ and those of $\sum_{x \in dom(\sigma)} G(x\sigma)$ that are reachable from $X(d)$, *i.e.* those of $G(r')$, up to a renaming. The unfolding of H from its root therefore yields r', QED.

In the above theorem, g and d are supposed to be linear terms. If d is not linear the rewriting of $G(r)$ yields a hypergraph H with more sharing than $G(r')$. If g is not linear, then r can rewrite into r' without having an occurrence of $G(g)$ in $G(r)$: our definition of graph rewriting does not allow to quotient the target graph for finding an instance of $G(g)$. In any case, if G is a graph that rewrites into H by a rule deduced from a term rewrite rule then exp(G) itself rewrites into the unfolding of H, perhaps in several steps. This is stated in the following theorem:

Theorem 11. *Let $g \to d$ be a term rewriting rule where neither g nor d is reduced to a variable. If $H \to K$ for an acyclic graph H of root s under the deduced graph rewriting rule then $\exp(H, s) \to^+ \exp(K, s)$ under $g \to d$.*

Proof: We may assume that H contains only vertices reachable from s, discarding the other vertices if necessary, and that it is not empty, since otherwise it cannot be rewritten. Since H rewrites into K by $G(g) \to G(d)$, we have the following decomposition, up to a vertex renaming:

$$H = C + G(g)$$
$$K = C + G(d)$$

for some context graph C. If C is empty then $H = G(g)$ and $K = G(d)$: the result is straightforward. Else, C is not empty and rooted by s. Let $X(g) = \{x_1, \ldots, x_n\}$ for $n \geq 0$ be the variables of g, and denote by $y, y_1, \ldots, y_n$ the renamings in H of vertices $x_\epsilon, x_1, \ldots x_n$ of $G(g)$ and by $G_1, \ldots, G_n$ the sub-hypergraphs of H reachable from $y_1, \ldots, y_n$. We have a further decomposition of C, in which C' collects all the arcs that are not included in $G(g) \cup G_i$:

$$C = C' + \textstyle\sum_{1 \leq i \leq n} G_i$$

Note that the G_i are not necessarily disjoint and that arcs in C' can have several references to $y, y_1, \ldots, y_n$. Note also that none of the G_i references u since otherwise H would not be acyclic. Since H is acyclic, so are its subgraphs $G(g)$, C' and the G_i, which can therefore be unfolded. Let t_i, $1 \leq i \leq n$, be the unfolding of G_i starting from y_i, and c the unfolding of C' when considering y as a (term) variable: This amounts to discard $G(g)$ for the unfolding of C':

$$\begin{aligned} c &= \exp(C' + \sum_{1 \leq i \leq n} G_i, s) \\ &= \exp(C', s)[\exp(G_1, x_1)/x_1, \ldots, \exp(G_n, x_n)/x_n] \\ &= \exp(C', s)[t_1/x_1, \ldots, t_n/x_n] \end{aligned}$$

where $v[t_1/x_1, \ldots, t_n/x_n]$ stands for the simultaneous substitutions of terms t_i to all the occurrences of x_i in v. For H and K, we have the following unfoldings:

$$\begin{aligned} \exp(H, s) &= \exp(C' + \sum_{1 \leq i \leq n} G_i + G(g), s) \\ &= \exp(C' + \sum_{1 \leq i \leq n} G_i, s)[\exp(G(g) + \sum_{1 \leq i \leq n} G_i, y)/y] \\ &= c[\exp(G(g) + \sum_{1 \leq i \leq n} G_i, y]/y] \\ &= c[g[t_1/x_1, \ldots, t_n/x_n], y] \end{aligned}$$

and similarly:

$$\begin{aligned}
\exp(K, s) &= \exp(C' + \sum_{1 \le i \le n} G_i + G(d), s) \\
&= \exp(C' + \sum_{1 \le i \le n} G_i, s)[\exp(G(d) + \sum_{1 \le i \le n} G_i, y)/y] \\
&= c[\exp(G(d) + \sum_{1 \le i \le n} G_i, y)/y] \\
&= c[d[t_1/x_1, \ldots, t_n/x_n], y]
\end{aligned}$$

Since all the occurrences of $g[t_1/x_1, \ldots, t_n/x_n]$ in exp(H, s) are independent of one another, this proves that exp(H) rewrites in exp(K) by parallel applications of the rule $g \to d$ to the distinguished instances of g in exp (H), QED.

A comparison of term rewriting and graph rewriting: Term rewriting can in general be implemented with our set-theoretic graph rewriting, as it is the case for other graph rewritings. This allows to simulate several steps of term rewriting using a single step of graph rewriting. In this approach, the only vertices that matter are those reachable from the given root of the graph. Rewriting with rules having different variables in their left and right-hand sides yields graphs with extra vertices, that should be disposed of by a garbage-collector, that is, an external mechanism. This is not a problem, since garbage collectors have existed for a long time and are also needed by term rewriting engines. This phenomenon is due to the non-reversibility of the rules (cf. Lafont [1990]). No such problem occurs with when we consider graphs on their own, not as representative for terms. In this case, there is no reason to discard any vertex, nor to have same sets of variables in both sides of a rule.

One problem could be the impossibility to handle "collapsing rules" of a general form $t[x] \to x$, where x is a variable: Our definition does not allow a term to be collapsed to a variable. This reflects our view that rewriting is not the same thing as computing quotients: collapsing rules achieve a form of quotient since they have a global effect on the target graph whenever t is shared. Rewriting requires to know all predecessors of t: this takes a traversal of the whole graph before replacing the left-hand side. In our approach, these rules could be simulated by their embedding in all elementary contexts: add to the rewriting system all the rules schema $G(fw_1t[x]w_2) \to G(fw_1xw_2) + G(t(x))$ for all function symbol f and words w_1, w_2 in X^* having lengths compatible with the arity of f! A more practical way is to consider $t[x] \to x$ as being $t[x] \to Ix$, where I is a new symbol, and adding the rule schema for I solely: $fw_1yw_2 + Iyx \to fw_1xw_2 + Iyx$! This amounts to use indirections, a standard way to implement erasing rules on conventional systems. Erasing is postponed until it is actually needed.

V. CONFLUENCE

In the case of terms, there is a criterion for confluence (cf. Knuth & Bendix[1970]). There is a similar criterion in the case of graphs, which is actually easier to prove. We shall need the following lemma of transitivity.

Lemma 12. *If* $G \to D$ *generates* $G_1 \to D_1$ *and* $G_1 \to D_1$ *generates* $G_2 \to D_2$, *then* $G \to D$ *generates* $G_2 \to D_2$.

Proof: Provided a suitable renaming of variables of $G_1 \to D_1$ and $G \to D$, both hypotheses amount to:

$$G_1 = C_1 + G,$$
$$D_1 = C_1 + D,$$
$$X(G) \cap X(C_1) \supseteq X(D) \cap X(C_1),$$

and

$$G_2 = C_2 + G_1,$$
$$D_2 = C_2 + D_1,$$
$$X(G_1) \cap X(C_2) \supseteq X(D_1) \cap X(C_2).$$

Since $X(G_1) = X(G) + X(C_1)$ this last inequality implies

$$X(G) \cap [X(C_2) - X(C_1)] \supseteq X(D) \cap [X(C_2) - X(C_1)]$$

Then by associativity of the set-union, we have:

$$G_2 = C_2 + C_1 + G$$
$$D_2 = C_2 + C_1 + D$$

and we only have to check the inequality on variables:

$$\begin{aligned} X(G) \cap [X(C_1) + (X(C_2) - X(C_1))] &= X(G) \cap X(C_1) + X(G) \cap [X(C_2) - X(C_1)] \\ &\supseteq X(D) \cap X(C_1) + X(D) \cap [X(C_2) - X(C_1)] \\ &= X(D) \cap [X(C_1) + (X(C_2) - X(C_1))] \end{aligned}$$

QED.

Recall the following standard definition.

Definition 13. *A relation $\to$ is locally confluent if for all pairs $C \leftarrow A \to B$ there exists some graph D such that $C \to^* D \leftarrow^* B$.*

In this definition, should B and C rewrite to the same graph D, or to two isomorphic graphs? The following example explores both possibilities.

Example 9: consider the following two rules: $ax \leftarrow ax + ay \to ay$. Are the two right-hand sides equal? No, only isomorphic. Take a simple context and apply the two rules:

$$bxx + ax \leftarrow bxx + ax + ay \to bxx + ay$$

The two results are no longer isomorphic: graph isomorphism is not preserved by embedding in an arbitrary context; only graph equality is preserved (this is the first example of section II). However, graph equality is too strict for the new variables. For instance given the rule $ax \to ax + ay$, the graph ax rewrites to $ax + ay$, but also to $ax + az$. These are not equal, only isomorphic, but the renaming involves only the "new" variable y of the right-hand side of the rule. If a context is added to the rule, its variables will not be involved in the renaming, because of condition (*ii*) of Definition 2. This remark justifies the following extension of local confluence.

Definition 14. *A graph relation* $\rightarrow$ *is locally confluent if all pairs* $C \leftarrow A \rightarrow B$ *meet in the following sense: there exist two graphs* D_1 *and* D_2 *such that* $B \rightarrow^* D_1$ *and* $C \rightarrow^* D_2$ *and* D_1 *and* D_2 *are equal up to a variable renaming having domain disjoint from* $X(A)$.

In the last example, the renaming is given by $y \mapsto z$ and x is not renamed.

Theorem 15. *For a rewriting relation to be locally confluent it is necessary and sufficient that for all pairs* $G_1 \rightarrow_1 D_1$ *and* $G_2 \rightarrow_2 D_2$ *and all graphs* G *which are unions of* G_1 *and* G_2 *the pairs* $H_1 \underset{1}{\longleftarrow} G \underset{2}{\longrightarrow} H_2$ *meet.*

Proof: the necessary condition is trivial, since it is a particular case of the confluence property. To check the sufficient condition, note that if $C \leftarrow A \rightarrow B$ there exist two contexts H_1 and H_2 such that $A = H_1 + G_1$ is a disjoint union and $B = H_1 + D_1$, on one hand; and $A = H_2 + G_2$ is a disjoint union and $C = H_2 + D_2$ on the other hand. Therefore, setting $H = H_1 \cap H_2$ we have

$$A = H + G_1 + (G_2 - G_1) = H + G_2 + (G_1 - G_2) \; (= H + G_1 \cup G_2)$$
$$B = H + D_1 + (G_2 - G_1)$$
$$C = H + D_2 + (G_1 - G_2)$$

The hypothesis ensures the existence of two graphs K_1 and K_2 which are isomorphic via a renaming having a domain disjoint from $X(G_1 \cup G_2)$ and such that

$$D_1 + G_2 - G_1 \xrightarrow{*} K_1$$
$$D_2 + G_1 - G_2 \xrightarrow{*} K_2$$

These two relations generate (adding the context H)

$$H + D_1 + G_2 - G_1 \xrightarrow{*} H + K_1'$$
$$H + D_2 + G_1 - G_2 \xrightarrow{*} H + K_2'$$

where K_1' and K_2' are respectively isomorphic to K_1 and K_2 via two renamings having domains disjoint from $X(H + G_1 \cup G_2)$. The above lemma ensures that these two relations are indeed generated by the system of rules: they are derivations. Extending this isomorphism by the identity over $X(H)$ implies that the graphs $H + K_1'$ and K_2' are isomorphic, QED.

VI. CONCLUSION

The set-theoretic method of rewriting graphs has advantages and drawbacks. The main advantage is its simplicity: no categorical background, intuitive content. Left-hand sides are subtracted, right-hand sides are added. Overlapping simply amounts to non-empty intersection. The term-theoretic definition is not so simple. And in the Berlin approach, the role of intersection is held by pull-backs (this is necessary in the case of non-injective occurrences of left-hand sides).

The main drawback is the impossibility of handling quotients. This is not really a handicap for graph quotients identifying two vertices belonging to the left-hand side. In

the Berlin approach, one condition for applying a (fast) rewriting rule is a restriction on the identification of two vertices. It is more annoying for simple quotients like $Ix \to x$. But the implementation of such a rule is not so simple as its appearance: one has to look for all the "fathers" of the node labelled I and change their pointer to it. This implies visiting the whole graph, or keeping along with each node the list of all its fathers, as in Dactl (see Kennaway [1988]). Nevertheless, this is a drawback of the method. How to mend it still is an open problem.

Actually, our thesis is that graph manipulations are really of two sorts: rewritings, in which each step is simple, essentially an O(1) operation, and quotients, which are more complex operations, since the identification of two vertices sometimes entails visit of the whole graph. Our set-theoretic rewritings are restricted to be of the first type. They are free in some sense. Also, many such rewritings satisfy the extra condition of reversibility, which can easily be checked. This can be helpful, when dealing for instance with a data base in which one wants (sometimes) to be able to return to a previous state.

REFERENCES

Barendregt H, van Eekelen M., Glauert J., Kennaway R., Plasmeijer M. & Sleep R.[1987] : *Term Graph Rewriting*, Proc. PARLE Conference, Eindhoven, 1987, Lecture Notes in Computer Science 259, pp 141-158, Springer Verlag.

Bauderon M. & Courcelle B. [1987] : *Graph Expressions and Graph Rewritings*, Mathematical Systems Theory, Vol. 20, 1987, pp. 83-127.

Dershowitz N. [1985] : *Computing with rewrite Systems*, Information and Control, Vol. 64 (2/3), 1985, pp. 122-157.

Ehrigh H., Pfender M. & Schneider H-J. [1973]: *Graph Grammars: an algebraic approach*, Proc. 14th IEEE Symp. on Switching and Automata Theory, 1973, pp. 167-180.

Ehrig H. [1987]: *Tutorial Introduction to the Algebraic Approach of Graph Grammars*, 3rd Int. Workshop on Graph Grammars and their Applications to Computer Science , 1987, Ehrig, Nagl, Rozenberg & Rosenfeld (Eds), Lecture Notes in Computer Science 291, Springer Verlag, pp. 3-14.

Habel A & Kreowski H-J. [1987] : *May we introduce to you: hyperedge replacement*, 3rd Int. Workshop on Graph Grammars and their Applications to Computer Science , 1987, Ehrig, Nagl, Rozenberg & Rosenfeld (Eds), Lecture Notes in Computer Science 291, Springer Verlag, pp. 15-26.

Habel A & Kreowski H-J. [1987]: *Some structural aspects of hypergraph languages generated by hyperedge replacement*, Proc. STACS '87, Lecture Notes in Computer Science 247, pp. 207-219. Habel A, Kreowski H-J. & Plump D. [1987] : *Jungle Evaluation*, "Recent Trends in Data Type Specification", 5th Workshop on Specification of Abstract Data Types, Gullane (Scotland), 1987, Lecture Notes in Computer Science 232, pp. 92-112, Springer Verlag.

Handbook [1990] : *Handbook of Theoretical Computer Science*, Vol. B: Formal Methods and Semantics, J. van Leeuwen Ed., Elsevier 1990.

Kennaway R. [1987] : *On "On Graph Rewritings"*, Theoretical Computer Science 52, 1987, North-Holland, pp. 37-58; see also *Corrigendum to On "On graph Rewritings"*, Theoret-

ical Computer Science 61, 1988, North-Holland, pp. 317-320

Kennaway R. [1988] : *Implementing Term Rewrite Languages in DACTL*, Proc. of C.A.A.P. '88, 1988, Lecture Notes in Computer Science 299, Springer Verlag, pp. 102-116.

Kirchner C. [1985] : *Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories équationnelles*, Thèse d'Etat, Univ. Nancy, 1985.

Knuth D.E. & Bendix P.B. [1970] : Simple word problems in universal algebras, *in Computational Problems in Abstract Algebra*, Leech J. ed., Pergamon Press, Braunschweig (1970).

Lafont Y. [1990] : *Interaction Nets*, Principles of Prog. Languages, San Francisco, 1990, pp 95-108.

Löwe M. & Ehrig H. [1990] : *Algebraic Approach to Graph Transformation Based on Single Pushout Derivations*, Proc. 16th Int. Workshop on Graph Theoretic Concepts in Computer Science (WG'90), R. H. Möhring, Lecture Notes in Computer Science 484 (1990).

Muller D. & Schupp P. [1985] *The theory of ends, pushdown automata, and second order logic*, Theoretical Computer Science 37, pp 51-75.

Nagl M. [1979] : *A Tutorial and Bibliographical Survey on Graph Grammars*, 1st Int. Workshop on Graph Grammars and their Applications to Computer Science and Biology, 1979, Ehrig, Claus & Rozenberg (Eds), Lecture Notes in Computer Science 79, Springer Verlag.

Raoult J-C. [1984]: *On Graph Rewritings*, Theoretical Computer Science 32, 1984, North-Holland, pp. 1-24.

Raoult J-C. [1985]: private communication.

Rusinowitch M. [1989] : *Démonstration Automatique: techniques de réécriture*, InterEditions, Paris, 1989.

Sopena E. [1987] : *Combinatorial hypermap rewriting*, in Proc. RTA 87, Bordeaux, Lecture Notes in Computer Science 256 pp. 62-73.

PI 640 TOWARDS THE RECONSTRUCTION OF POSET
Dieter KRATSCH, Jean-Xavier RAMPON
Mars 1992, 22 pages.

PI 641 MADMACS : A TOOL FOR THE LAYOUT OF REGULAR ARRAYS
Eric GAUTRIN, Laurent PERRAUDEAU
Mars 1992, 12 pages.

PI 642 ARCHE : UN LANGAGE PARALLELE A OBJETS FORTEMENT TYPES
Marc BENVENISTE, Valérie ISSARNY
Mars 1992, 132 pages.

PI 643 CARTESIAN AND SATISTICAL APPROACHES OF THE SATISFIABILITY PROBLEM
Israël-César LERMAN
Mars 1992, 58 pages.

PI 648 SET-THEORETIC GRAPH REWRITING
Jean-Claude RAOULT, Frédéric VOISIN
Mars 1992, 18 pages.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique

ISSN 0249-6399